Лабораторна робота №3

Тема: "PHP: створення бази даних. Створення таблиць. Заповнення таблиць. Створення запитів БД."

Мета: Набуття навичок створення та редагування БД, таблиць.

Теоретичні відомості

Скриптова мова PHP надає потужний і простий механізм взаємодії WEB-сервера і SQL-сервера. Власне кожен PHP-скрипт може запускати стандартні команди управління SQL-сервера. Для нашої подальшої роботи спочатку необхідно ознайомитись із основними командами управління SQL-сервером:

1. Функція `include()`

Функція `include(filename);`

Функція `include()` виконує саме те, про що повідомляє її ім'я: вона включає ("врізає") в файл `filename` в склад того PHP-скрипту і в те його місце, де знаходиться ця команда.. Вона вельми зручна, оскільки дозволяє використовувати заздалегідь підготовлені фрагменти коду, які можуть бути вставлені в різні сторінки, але зберігаються в одному місці. Синтаксис функції такий:

```
include("file.inc");
```

Якщо ви розташуєте вище приведений рядок в одному із своїх сценаріїв, то зможете застосувати будь-які функції, які будуть визначені файлі **`file.inc`**. `Насправді` можна використовувати не тільки функції, але й будь-яку інформацію, що міститься у цьому файлі. Наприклад, в ньому можна надати значення змінним або написати фрагменти HTML коду. Файли, що включаються в сценарій, як правило, мають розширення **.inc**, що спрощує їх пошук.

2. Функція `mysql_connect()`

Функція `mysql_connect` – Відкриває з'єднання з сервером `MYSQL`.

Вона повертає покажчик на з'єднання з `MYSQL` у разі успішного виконання, або `FALSE` при в разі невдачі.

При її виклику, в якості параметра повинні бути вказані місцезнаходження сервера, реєстраційне ім'я і пароль користувача MYSQL. Значення, яке повертає `mysql_connect()`, зазвичай привласнюється змінній, яка потім використовується функцією `mysql_select_db()`. `Наприклад:`

```
$server_connection = mysql_connect ("localhost", "username ", "password ");
```

Localhost - ім'я сервера
Username - ім'я користувача власника процесу сервера
Password - пароль

З'єднання з сервером буде закрито при завершенні виконання скрипта, якщо до цього воно не буде закрито за допомогою функції **mysql_close().**

3. Функція mysql_select_db()

Функція mysql_select_db() - вибирає базу даних на SQL-сервері для подальшої роботи. Наприклад:

```
mysql_select_db("news", "$server_connection");
```

Ця функція дорівнює TRUE у разі успішного завершення або FALSE у разі виникнення помилки. В нашому прикладі успішність її виконання ми не контролюємо, адже значення цієї функії ні якій змінній не присвоюється.

4. Функція mysql_query()

Функція mysql_query() - надсилає запит до SQL-сервера. (Дуже важлива функція!) Зміст самого запита треба вказувати в дужках.

Якщо в якості запитів використовувались команди **SELECT, SHOW, EXPLAIN, DESCRIBE,** тоді mysql_query() буде дорівнювати результату виконання цього запиту (точніше сказази – показчику на цей результат). Ну і традиційно – FALSE, якщо запит не був виконаний.

У решті випадків, mysql_query() дорівнює TRUE у разі успішного запиту і FALSE у разі помилки. Наприклад:

```
$result = mysql_query ("SELECT * FROM news");
```

В змінній **$result** буде знаходитись результат виконання команди **SELECT** до таблиці **news.** Якщо забігти наперед, то в ній опиниться масив із стрічок. Причому їх буде стільки, скільки було витягнуто стрічок із таблиці **news.**

5. Функція mysql_fetch_array

Функція mysql_fetch_array дозволяє послідовно витягати по ожному рядку із змінних типу **$result** - результату виконання запитів **mysql_query**.

При кожному виклику цієї функції повертається наступний рядок даних з результатів запиту до тих пір, поки не будуть оброблені всі рядки. На масив можна посилатися по номеру або по імені стовпця:

```
$myrow = mysql_fetch_array($result);
```

5. Запити SQL

Увага ! *Нижче ми розглянемо основні SQL-запити. Зауважимо, що далі у нас йдуть "чисті" SQL-команди, а не ті, які формує мова PHP (наприклад, такі ми щойно вище розглянули). Але, в той же час, ці команди можна безпосередньо використовувати в складі команд PHP.*

Як правило, при розробці програм SQL прийнято вводити ключеві слова прописними літерами. Наприклад - SELECT, WHERE та UPDATE. В результаті запити SQL стануть зручнішими для читання

Ключове слово	Визначення та приклад
SELECT	Застосовується для вибірки даних з бази даних SELECT * FROM news WHERE news_id > '1'
INSERT	Застосовується для вставки нової інформації в базу даних INSERT INTO news VALUES (NULL, '1', '11-16-2006', 'Загоруйко В.')
DELETE	Застосовується для видалення записів з бази даних DELETE FROM news WHERE news_id = '5'
UPDETE	Дозволяє змінити дані в базі даних UPDATE news SET urgent ='YES' WHERE news_id='5'

Типи полів таблиць баз даних

Пам`ятаємо, що бази даних складаються з окремих таблиць. В той же час кожна таблиця має окремі поля. Поля, як і звичайні змінні бувають різних типів.

Список типів, що найчастіше зустрічаються, приведений в таблицях 3.1 - 3.3. Для багатьох типів даних задається максимальна ширина відображення, що вказується в дужках, яку ми далі позначатимемо символом max. Наприклад, запис INT(2) означає, що значення даного поля не може перевищувати 100.

До числових типів відносяться цілі числа і числа з плаваючою крапкою. Для чисел з плаваючою крапкою, окрім максимальної ширини відображення можна також указувати число значущих цифр після коми, що далі позначається символом P.

Таблиця 3.1. Числові типи

Тип	**Опис**
TINYINT[(max)]	Дуже маленькі цілі числа діапазону -127…128.
SMALLINT[(max)]	Маленькі цілі числа діапазону -32768…32767.
MEDIUMINT[(max)]	Середні цілі числа.
INT[(max)]	Звичайні цілі числа.
FLOAT[(max,P)]	Числа з плаваючою точкою одинарної точності.
DOUBLE[(max,P)]	Числа з плаваючою точкою подвійної точності.

DECIMAL[(max,P)]	Числа з плаваючою крапкою, приведені до типу char.

Типи дати і часу приведені в таблиці 3.2.

Таблица 3.2. Типи дати і часу

Тип	**Опис**
DATE	Дата у форматі РРРР-ММ-ДД.
TIME	Час у форматі ЧЧ-ММ-СС.
DATETIME	Дата и час у форматі РРРР-ММ-ДД ЧЧ-ММ-СС.
YEAR	Рік у формате РР або РРРР.
TIMESTAMP	Мітка часу для відліків по транзакціях у форматі РРРР-ММ-ДД ЧЧ-ММ-СС.

Основні **рядкові типи** приведені в таблиці 3.3.

Таблиця 3.3. Рядкові типи VARCHAR(len)[BI

Тип	**Опис**
CHAR (len) [BINARY]	Рядки з довжиною в len-символів. Не мають перевищувати 255 символів. Ключове слово BINARY указує на те, що дані повинні оброблятися незалежно від регістра.
CHAR	Синонім CHAR(1).
VARCHAR(len) [BINARY]	Синонім CHAR(len) за винятком того, що рядки можуть бути довільної довжини.
TEXT	Рядки з максимальною довжиною символів рівної 65535. Дані цього типу чутливі до регістра.
BLOB	Двійкові рядки з максимальною довжиною символів рівної 65535. Тип BLOB (binary large object - великий двійковий об'єкт) призначений для зберігання двійкових даних, зокрема зображень і звукових послідовностей.

Створення простої бази даних

Для створення бази даних треба виконати таку команду, вказавши замість DBNAME реальне ім’я БД:

```
CREATE DATABASE dbname;
```

Кожен запит MySQL завершується крапкою з комою.

Для того, щоб переконатися, що база даних успішно створена, можна виконати ко

манду SHOW DATABASES, яка покаже, які бази даних існують на вашому комп'ютері:

```
mysql> SHOW DATABASES;
```

USE

Для того, щоб почати роботу з таблицями, необхідно повідомити MySQL з якою базою даних ви маєте намір працювати. Це здійснюється за допомогою команди USE:

```
mysql> USE dbname;
```

Тут dbname - назва вибираної бази даних.

Сервер повинен повернути таку відповідь:

```
Database changed;
```

CREATE TABLE

Тепер можна створити таблицю, яка буде використовуватись в подальшому проекті:

CREATE TABLE table_name

Тут table_name – ім'я створюємої таблиці.

```
mysql> GREATE TABLE news (
-> news_id INT NOT NULL AUTO_INCREMENT,
-> heading VARCHAR (48),
-> body TEXT,
-> data DATE,
-> author_name VARCHAR (48),
-> author_email VARCHAR (48),
-> PRIMARY KEY (news_id));
```

Первинному ключеві (PRIMARY KEY) можна привласнити атрибут auto_increment, що дозволяє автоматично генерувати унікальний ключ, якщо його тип є цілочисельним. При вставці запису в базу даних значення ключа спочатку виставляється рівним нулю, а далі MySQL автоматично буде збільшувати його значення на одиницю.

Основні типи запитів SQL

Побудувавши першу таблицю, можна попрактикуватися в складанні та виконанні SQL-запитів. Подібні вправи допоможуть вам вивчити найбільш часто вживані запити SQL.

Перед тим , як вводити данні у таблицю розгляньте її структуру. Для цього скористайтеся командою **DESCRIBE.**

DESCRIBE

Команда **DESCRIBE** показує структуру створених таблиць і має наступний синтаксис:

DESCRIBE table_name

Тут table_name - ім'я таблиці структура якої запрошується.

Якщо ввести:
mysql> **DESCRIBE** news;
то сервер видасть опис таблиці news:

Field	Type	Null	Key	Default	Extra
news_id	int(11)		PRI	0	auto_increment
heading	varchar(48)	YES		NULL	
body	text	YES		NULL	
date	date	YES		NULL	
author_name	varchar(48)	YES		NULL	
author_email	varchar(48)	YES		NULL	

6 rows in set (0.00 sec)

Короткий опис елементів таблиці:

- Кожна стрічка в графі Field повідомляє ім'я поля (стовпця таблиці), в якому містяться дані.
- В графі Type поміщена інформація про те, які дані і якого типа можуть зберігатися в кожному конкретному полі таблиці.
- Якщо в графі NULL відповідного поля стоїть слово YES, то значення цього поля може бути рівним NULL.
- Наявність напроти конкретного поля в графі Key позначення PRI (скорочення від слова primary – первинний ключ) указує на те, що дане поле є індексним при сортуванні таблиці. У стовпці первинного ключа не можуть знаходитися два однакових значення.
- Графа Default указує на те, які значення вводяться в це поле за замовчанням.
- І, нарешті Extra, описує будь-які додаткові атрибути, які може мати відповідна графа таблиці. Наприклад, атрибут auto_increment, призначений первинному ключу, указує, що збільшується на 1 кожного разу при додаванні нового рядка в таблицю.

Введення даних

Тепер помістимо в таблицю деякі дані. Якщо ввести наступну інформацію за підказкою Mysql:

```
Mysql> INSERT INTO news
-> VALUES (NULL, 'Назва','Текст новин',
-> '11-16-2006','Загоруйко В.',
-> 'zagor@domain.com');
```

То сервер повинен повернути нам таку відповідь:

```
Query OK, 1 row affected (0.00 sec)
```

Направимо запит до бази даних за допомогою команди **SELECT** і узнаємо, як ця інформація виглядає у вигляді таблиці. У вказаній команді оператор * діє як звичний символ-заступник UNIX.

Якщо ввести в командному рядку MySQL:

```
Mysql> select * from news;
```

Відповідь сервера повинна бути наступною:

News_id	Heading	Body	Date	Author_name	Author_email
1	Назва	Текст новин	0000-00-00	Загоруйко В	zagor@domain.com

```
1 row in set (0.06 sec)
```

Заміна (корегування) даних в таблиці

Що робити коли ви неправильно ввели дані і хочете їх відкоригувати? Продивимось приведені вище результати. Ви помітили помилку? Дата роздрукована у вигляді 0000-00-00. Помилка виникла тому, що при введені дати був використаний невірний формат. Ми намагалися указати дату у вигляді 11-16-2006, а потрібно було – 2006-11-16. Поле дати в базі даних MySQL не приймає значення, яке не відповідає обов'язковим критеріям. А по цих критеріях дата повинна бути представлена як номер року, що складається з чотирьох цифр, за яким слідує дефіс, потім - місяць, позначений двома цифрами, знову дефіс, потім – день, представлений двома цифрами.

Щоб виправити помилку, можна скористатися командою **UPDATE.** Вводимо в командному рядку MySQL:

```
Mysql> update news set date='1006-08-16' where
news_id='1';
```

Сервер повинен повернути таку відповідь:

```
Query OK, 1 row affected (0.03 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

Щоб переконатись у виконанні всіх необхідних виправлень, ще раз укажіть команду **SELECT:**

```
Mysql> select * from news;
```

Відповідь сервера повинна бути наступною:

News_id	Heading	Body	Date	Author_name	Author_email
1	Назва	Текст новин	2006-11-16	Загоруйко В.	zagor@domain.com

```
1 row in set (0.06 sec)
```

Як видно з цих результатів, тепер в базі даних зберігається вірна інформація.

Видалення даних. DELETE.

Для видалення стрічки таблиці треба ввести:

```
Mysql> delete from news where news_id='1';
```

Відповідь сервера має вигляд:

```
Query OK, 1 row affected (0.03 sec)
```

Знову виконаємо команду SELECT, щоб дізнатись, який вигляд має таблиця:

```
Mysql> select * from news;
```

Сервер надасть нам наступну відповідь:

```
Empty set (0.00 sec)
```

Сервер MySQL повернув повідомлення `Empty set` (порожній результуючий набір даних), оскільки жоден рядок в базі даних не відповідав критеріям, визначеним командою **SELECT**. В цьому випадку критерій вибірки даних (оператор *) указував на те, що повинні бути вибрані всі записи в таблиці. А оскільки повернений порожній набір, то в таблиці нічого немає. Порожній результуючий набір міг бути повернений і при використанні критерію вибірки в іншій формі (як, наприклад, в команді **SELECT** `* from news where news_id='5'`, хоча при цьому в таблиці могли б знаходитися дані). Скажімо, таблиця могла містити дані в рядках, для яких `news_id` дорівнює 2, 3 або 4. Але оскільки в нашому випадку застосувався запит "видати все, що знаходиться в таблиці", і нічого не було повернено, означає, що таблиця дійсно порожня.

Команда **DELETE** просто видаляє рядок (або рядки), відповідні вказаному критерію. У розглянутому випадку з таблиці видалений рядок, значення news_id якого дорівнювало 1. У таблиці можна було також, наприклад, видалити рядки, в яких поле Author_name містить значення "Загоруйко В."; виконання даного оператора могло викликати видалення декількох рядків. Введення одного оператора **DELETE** може привести до видалення сотні стрічок даних, якщо він, наприклад, оформлений у вигляді команди

```
delete from news where news_id > '0';
```

Хід виконання роботи

І частина

1. Побудувати сценарій, який буде заносити новини у БД.

Введення інформації в базу даних з використанням PHP

Для роботи з базою даних застосування PHP є менш трудомістким, гнучкішим і, що найважливіше, легше здійсненним за допомогою Web-браузера.

У основі взаємодії PHP та бази даних покладено прості принципи:

- спочатку необхідно приєднатись до серверу БД та зареєструватись;
- потім слід вибрати базу даних, яка стане використовуватися;
- і, нарешті, - відправляти запити SQL на сервер для додавання, видалення і зміни даних.

Вам навіть не треба пам'ятати про те, що в кінці роботи потрібно закрити з'єднання з сервером. PHP виконає це за вас.

Зараз ви повинні побудувати таблицю `news` за допомогою PHP. Для цього треба створити два додаткових файла.

1.1 Файл `inc.php`, який треба приєднати до сценарію, використовуючи функцію **`include`**, роботу якої ми розглянули раніше:

```
<?php
$cServname = "ім'я сервера";
$cUsername = "ім'я користувача БД";
$cPassword = "пароль користувача БД";
$cDatabase = "назва БД";
?>
```

У цьому сценарії ми запам'ятовуємо в змінні всі данні, які далі будемо використовувати для підключення до серверу БД. **Цей файл вже створено, вам потрібно буде тільки приєднати його до вашого головного сценарію.**

```
include('../inc.php');
```

1.2 Створюємо файл `inc_tab.php`, який теж приєднується до сценарію, за допомогою функції **`include`**:

```
<?
$cTablename1 = 'news';
?>
```

Ви бачите, що тут створюється змінна `$cTablename1`, яка запам'ятовує назву таблиці. **Кожен студент повинен надати свою індивідуальну назву таблиці! Використайте для її назви свій `login`.**

1.3 Створюємо сценарій `news.php`:

Наша таблиця Новин повинна мати такі поля:

- Порядковий номер - `news_id;`
- Заголовок новини – `heading;`
- Зміст новини – `body;`
- Дату створення – `date;`
- Ім'я автора - `author_name;`
- Email автора - `author_email.`

Типи цих полів, їх розмір та сама структура таблиці була розглянута раніше. Також ви

тут бачите застосування функцій, розглянутих на початку лабораторної роботи.

```
<?php

include('../inc.php'); //підключаємо файл з інформацією про підключення
include('inc_tab.php'); // підключаємо файл з інформацією про таблиці

$db = mysql_connect($cServname,$cUsername,$cPassword) or die('Не можу приєднатись до MySQL сервера.');
// підєднатись до MySql сервера
// або вивести повідомлення про неможливість приєднання

if (!mysql_select_db( $cDatabase, $db ))//якщо неможливо вибрати базу даних (тобто її не існує)то
{
     mysql_query( "CREATE DATABASE `$cDatabase`", $db ); // ми її створюємо.
}
mysql_select_db( $cDatabase, $db ) or die ("Не можу приєднатись до бази даних.");

// створюємо таблицю News, яка збирає замітки новин і має такі поля
mysql_query("CREATE TABLE `$cTablename1`(
     news_id INT NOT NULL auto_increment,
     heading VARCHAR(48),
     body TEXT,
     date DATE,
     author_name VARCHAR(48),
     author_email VARCHAR(48),
     PRIMARY KEY(news_id))",
     $db) or die(mysql_error()); //якщо запит не виконано - виводимо зміст помилки
print "ВІТАЮ!!!<br>";
echo"<h4>Таблиця $cTablename1 успішно створена.</h4>";
mysql_close($db);
?>
```

1.4 Створюємо сценарій `database.php`, який буде заносити новини у БД. Цей сценарій демонструє, наскільки легко розмістити інформацію в БД, використовуючи PHP. Достатньо ввести лише декілька коротких стрічок. З тридцяти стрічок коду в цьому сценарії, лише біля десяти відносяться до PHP. У інших звичайна розмітка HTML. Зразки форм, які з'являються при виконанні сценарію, показані на рис. 3.1 та 3.2

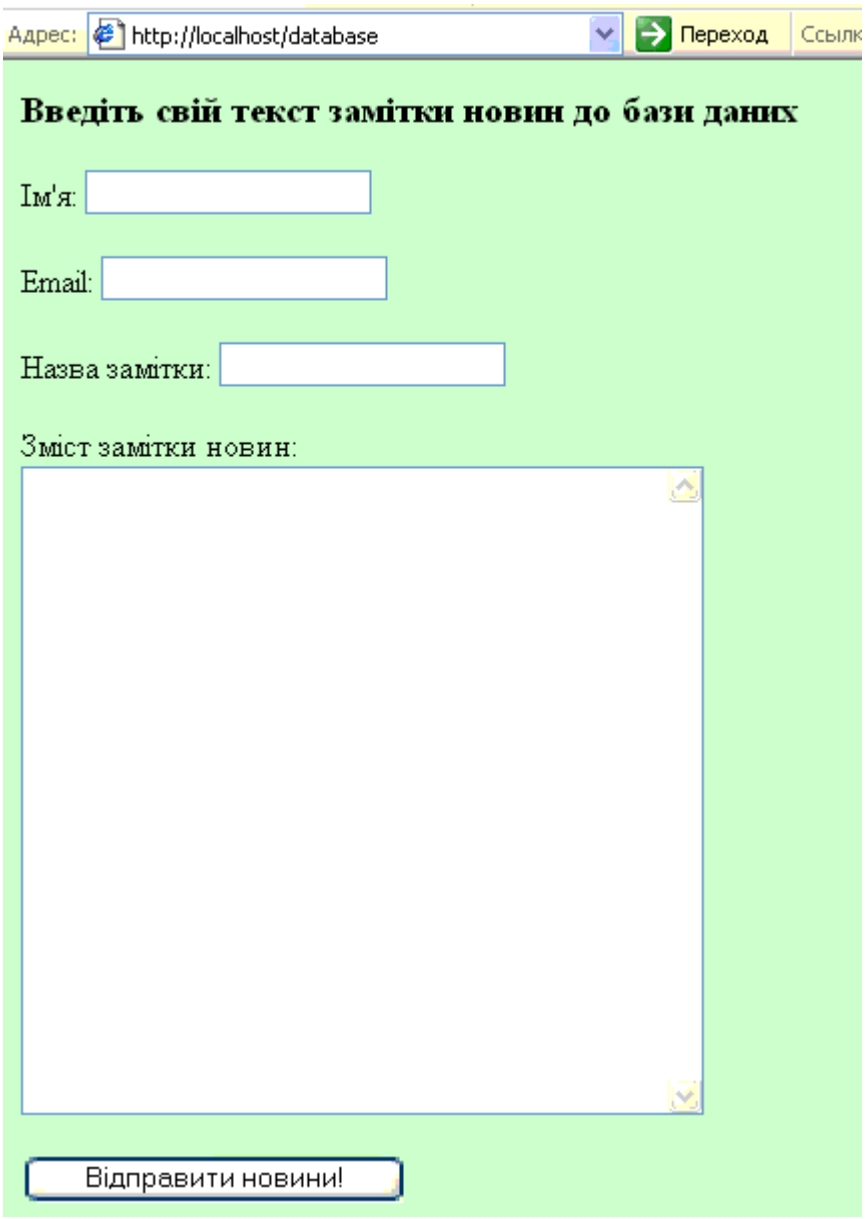

Рис. 3.1 Початкова форма введення інформації про новини в таблицю `news` з сценарію `database.php`

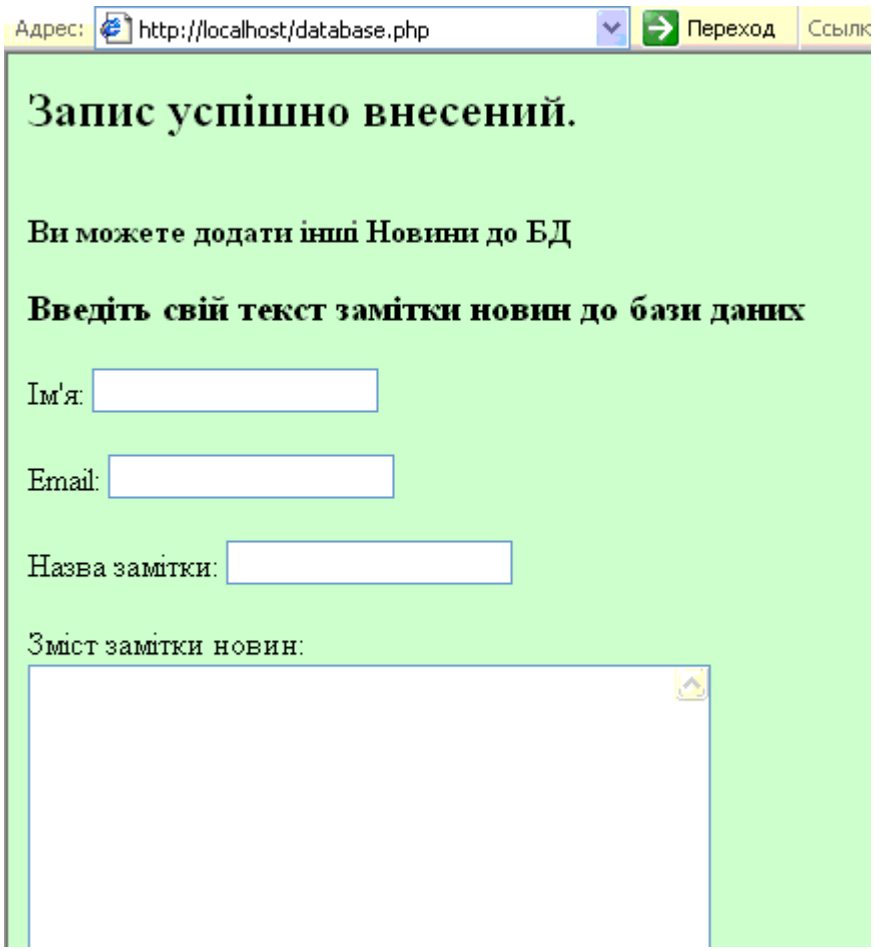

Рис. 3.1 Результат надсилання до сервера даних з форми при виконанні сценарію `database.php`

А це текст самого сценарію `database.php` **(Увага! При набиранні цього скрипту рядки не нумерувати!):**

```
1. <html>

2. <head>
3. <title>Сценарій відправлення новин на сервер БД</title>
4. </head>

5. <body bgcolor="#C4FFC4">

6. <?php

7. include('../inc.php');
8. include('inc_tab.php');

9. /* Ця програма дозволяє додавати Новини у базу даних */

10.  if (isset ($submit)): //якщо була нажата кнопка на формі
```

```
11.  $db     =     mysql_connect($cServname,$cUsername,$cPassword)or
   die('Не можу приєднатись до MySQL сервера.');
12.  mysql_select_db( $cDatabase, $db ) or die ("Не можу приєдна-
   тись до бази даних $cDatabase");


13.  $date=date("Y-m-d");
14.  $sql="INSERT INTO $cTablename1 VALUES
       (NULL,'$heading','$body','$date','$auth','$auth_email')";
15.  mysql_query($sql);

16.  echo "<h2>Запис успішно внесений.</h2>";
17.  print ("<br><b>Ви можете додати інші Новини до БД</b>");

18.  endif;
19.  ?>

20.  <p><h3>Введіть свій текст замітки новин до бази даних</h3></p>

21.  <form action="database.php" method="post">

22.  Ім'я: <input name="auth" type="text" ><p></p>
23.  Email: <input name="auth_email" type="text"><p></p>
24.  Назва замітки: <input name="heading" type="text"><p></p>
25.  Зміст замітки новин: <br>
26.  <textarea name="body" rows=20 cols=40></textarea><p></p>

27.  <input type="submit" name="submit" value="Відправити новини!">

28.  </form>
29.  </body>

30.  </html>
```

Пояснення до сценарію `database.php`

В стрічці 7 під'єднуємо файл `inc.php`, в якому налаштована вся інформація для з'єднання з сервером БД.

В стрічці 8 приєднуємо файл `inc_tab.php`, в якому зберігається назва таблиці.

В стрічці 10 перевіряється чи була натиснута кнопка `Submit`. Якщо так, то сценарій виконується. Якщо ні, відбувається перехід до частини `endif`, при цьому код сценарію PHP повністю ігнорується. Це дозволяє пришвидшити початкове завантаження сторінки.

В стрічці 11 результат виконання функції `mysql_connect()`, привласнюється змінній `$db` Ця змінна використовується у наступній стрічці для фактичного встановлення

з'єднання. Параметри даної функції указують сценарію, що повинно відбутись підключення до серверу MySQL.

В стрічці 12 відбувається виклик функції `mysql_select_db`. Вона дозволяє обрати для використання конкретну базу даних, а також ініціювати з'єднання з сервером шляхом виклику значення, вказаного в змінній `$db`.

У стрічці 14 змінній привласнюється значення стрічки, яка містить SQL-запит. Це робиться з ціллю спрощення коду виклику функції `mysql_query()`.

У стрічці 15 міститься виклик функції `mysql_query()`,яка відправляє оператор SQL на сервер MySQL. Цей оператор наказує серверу внести значення, отримані з форми у таблицю БД. В даному сценарії передбачено виведення повідомлення користувачу про те, що введенні їм дані були прийняті.

У стрічці 20 відбувається повернення до старого доброго коду HTML.

Цей сценарій викликає сам себе (стрічка 21), тому в одному сеансі може бути введено декілька заміток з новинами. Форма HTML виводиться на екран при кожному виконанні сценарію, та після чергового натиснення користувачем кнопки `Submit` в базу даних поступає нова інформація.

2. Для того, щоб продивитись зміст таблиці у БД, напишіть наступний сценарій:

Тут оператор `While` необхідний, оскільки невідомо наперед, який об'єм даних буде повернений запитом `SQL`. Один і той же запит може повернути один рядок даних, а може і п'ятдесят. Оператор `While` дозволяє пройти за всіма даними, поверненими після виконання запиту, і зупинитися, коли вони закінчаться.

```
<html>

<head>
<meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
 <title> зміст таблиці БД </title>
</head>

<body bgcolor=#FF8080 >

<?php

include('../inc.php');
include('inc_tab.php');

 /* Ця програма дозволяє продивитись зміст таблиці Новин у базі даних */
```

```
  $db  =  mysql_connect($cServname,$cUsername,$cPassword)or  die('Не
можу приєднатись до MySQL сервера.');
 mysql_select_db( $cDatabase, $db ) or die ("Не можу приєднатись до
бази даних $cDatabase");

 $date=date("Y-m-d");

 echo "<h2>Таблиця $cTablename1 $date</h2>";

 /* Тут запам'ятовуємо у змінну $rezult запит до серверу */
 $rezult= mysql_query("SELECT * FROM $cTablename1");

/* Змінна $items приймає перший рядок результату виконання MYSQL і
поміщає його в масив. Щоб доступитись до елементів масиву використо-
товуємо цикл while */
 while($items = mysql_fetch_row($rezult))

/* Виводимо результат у вигляді таблиці на екран, де у кожну комір-
ку прописуємо значення елементу масиву за його номером */

{
echo"<table border=1> <tr align=center>
 <td width=20>$items[0]</td>
 <td width=60>$items[1]</td>
 <td width=500>$items[2]</td>
 <td width=100>$items[3]</td>
 <td width=80>$items[4]</td>
 <td width=80>$items[5]</td>
 </tr></table>";
}
?>
</body>
</html>
```

3. Самостійно додати засоби контролю помилок, описані у попередній роботі.

4. Додайте 8-10 записів до БД.

5. Самостійно напишіть сценарій який буде видаляти записи БД за порядковим номером, та виводити результат на екран.

II частина

Вдосконалимо сценарій, який виводить зміст БД.

```
1. <html>
```

```
2. <head>
3. <meta      http-equiv="Content-Type"      content="text/html;
   charset=windows-1251">

4. <title> НОВИНИ</title>
5. </head>

6. <body bgcolor=#FFFFFF>
7. <h1><font  color=#0000A0>  Щоденні новини </font> </h1>

8. Сортувати статті за   


9. <a href="data_out.php?orderby=date">Дата</a>  

10. <a href="data_out.php?orderby=heading">

     Заголовок новини</a>  

11.     <a href="data_out.php?orderby=author">Ім'я автора </a>

12.     <p>
13.     <form action="data_out.php" method="post">

14.     Показати статті (<i>введіть ім'я автора </i>):

        <input name="author" type="text">
15.     <input type="submit" name="submit" value="Пошук">
16.     </form>
17.     </p>

18.     <table boreder="1" cellpading="3">

19.     <?php
20.     /* Ця програма дозволяє отримувати замітки з БД*/

21.     include('../inc.php');
22.     include('inc_tab.php');


23.     $db  =  mysql_connect($cServname,$cUsername,$cPassword)or
   die('Не можу приєднатись до MySQL сервера.');
24.     mysql_select_db( $cDatabase, $db ) or die ("Не можу при-
   єднатись до бази даних $cDatabase");

25.     if ($orderby=='date'):
26.     $sql="select * from $cTablename1 order by 'date'";
     elseif ($orderby=='author'):
     $sql="select * from $cTablename1 order by 'author_name'";
```

```
  elseif ($orderby=='heading'):
  $sql="select * from $cTablename1 order by 'heading'";
  elseif (isset($submit)):
  $sql="select * from $cTablename1 where author_name='$author'";

27.     else:

28.     $sql="SELECT * FROM $cTablename1";
29.     endif;

30.     $rezult= mysql_query($sql);

31.     while($row = mysql_fetch_array($rezult)){

32.     echo ("<tr><td bgcolor=#003399><b>");
33.     printf ("<font color=white>%s</font></b></td></tr>",
      $row["heading"]);
34.     printf ("<tr> <td>Від: <a href=mailto:%s>%s</a>",
      $row["author_email"], $row["author_name"]);
35.     printf ("<br>Дата: %s <hr>",
      $row["date"]);
36.     printf  (
37.     $row["body"],"%s</td></tr>");
38.     }
39.     ?>
40.     </body>
41.     </html>
```

Пояснення до сценарію

Як описувалося в попередній лабораторній роботі, URL в такій формі відправляє змінні в інтерпретатор PHP. Змінні, перераховані в рядках 9-11, використовуються в сценарії PHP для визначення того, які оператори SQL повинні піти на сервер MySQL.

Форма задана в рядках 13-16, застосовується для відправки на сервер необов'язкового запиту на відображення тільки заміток з новинами, написаних одним автором. Якщо користувач вводить ім'я автора, якого немає в базі даних, буде повернений порожній набір записів.

У рядках 21-24 міститься код, що встановлює з'єднання з сервером MySQL і вибирає базу даних, яка буде використовуватись для виконання запитів.

Рядки 25-27 надають користувачам певні можливості вибору категорії інформації, яку вони хочуть одержати з бази даних, і способу відображення цієї інформації. Посилання перераховані в рядках 9-11, і форма задана в рядках 13-16, обумовлюють, який з операторів if використовуватиметься. Змінна orderby визначає спосіб відображення даних. Якщо її значення не задане і кнопка Submit не була натиснута, то як запит SQL застосовується рядок 28.

До моменту переходу до стрічки 30 запит вже повинно бути виконано і дані записані в змінній $rezult.

У рядку 31 функція mysql_fetch_array() вибирає рядок з даних, повернених запитом. Після кожного виконання циклу оператор while переходить до іншого рядка. Цикл

продовжується до тих пір, поки mysql_fetch_array() не обробить всі повернені запитом рядки.

У рядках 32-37 за допомогою функції printf() виводиться кожен запис з бази даних, повернених функцією mysql_fetch_array(). Посилання на стовпці в рядках даних виконується за ім'ям стовпця.

І в рядку 37 завершується цикл while після обробки всіх стрічок даних, повернених запитом SQL.

Контрольні запитання:

1. Яка інформація міститься у файлі inc.php?
2. Для чого існує функція include(). Де і в яких сценаріях вона використовується?
3. Яка інформація міститься у файлі inc_tab.php?
4. Яке призначення функції mysql_connect()?
5. Яке призначення функції mysql_select_db()?
6. Яке призначення функції mysql_select_db()?
7. Яке призначення функції mysql_query()?
8. Яке призначення функції mysql_fetch_array ()?
9. Які SQL-запити ви знаєте?
10. Наведіть приклад запиту SELECT у зроблених вами скриптів?
11. Наведіть приклад запиту DELETE у зроблених вами скриптів?
12. Наведіть приклад запиту INSERT у зроблених вами скриптів?
13. Для чого призначена команда UPDATE?
14. Для чого призначена команда DESCRIBE?
15. Які є типи полів бази даних?
16. Як створити базу даних?
17. Як показати всі БД?
18. Як вибрати потрібну БД?
19. Як створити таблицю?
20. Що таке первинний ключ?
21. Яким чином вводяться дані у таблицю?
22. Яке призначення сценарію database.php? Яка послідовність створення цього сценарію? Як він працює?
23. Яке призначення сценарію news.php? Як він працює?